UBA - FCEyN - Departamento de Computación

ALGORITMOS Y ESTRUCTURAS DE DATOS III
Trabajo Práctico Nº2

Primera entrega: 12-MAY-2010, hasta las 20:00 horas.
Segunda entrega: 02-JUN-2010, hasta las 20:00 horas.

Ver información general sobre los Trabajos Prácticos en la página de la materia en Internet.

- Desarrollar e implementar algoritmos para los problemas enumerados.
- Calcular el orden de la complejidad de cada algoritmo en el modelo uniforme.
- Aplicar cada algoritmo a un conjunto de instancias de entrada, midiendo para cada instancia el tiempo de ejecución o la cantidad de operaciones. Elegir las instancias de manera tal que se pueda apreciar el comportamiento de los algoritmos. Graficar para cada algoritmo el tiempo de ejecución o la cantidad de operaciones conforme crece el tamaño de la entrada.
- Leer los datos de entrada desde un archivo con el nombre `Tp2EjX.in`, donde `X` es el número del problema. Escribir los resultados en un archivo con el nombre `Tp2EjX.out`.
- Respetar los formatos de archivos que se indican en cada caso. Como regla general, los valores contenidos en cada línea de los archivos de entrada están separados entre sí por uno o más espacios en blanco, y pueden estar precedidos o seguidos por una cantidad arbitraria de espacios en blanco; asimismo, los valores contenidos en cada línea de los archivos de salida están separados entre sí por exactamente un espacio en blanco, y no son precedidos ni seguidos por ningún caracter dentro de la línea. Utilizar como ejemplo los archivos provistos.
- Ignorar los costos de lectura y escritura de los archivos tanto al calcular complejidad como al medir.

1. Una secuencia de números es unimodal si es creciente hasta cierto elemento, y decreciente a partir de ese elemento. Por ejemplo, las secuencias $(1, 2, 5, 3)$, $(1, 2)$, $(5, 3)$ y (2) son unimodales, mientras que las secuencias $(2, 1, 2)$ y $(2, 2)$ no lo son. Toda secuencia puede transformarse en unimodal eliminando una cantidad suficiente de sus elementos, ya que una secuencia formada por un único elemento es siempre unimodal.

   Sea $s = (s_1, s_2, \ldots s_n)$ una secuencia de números enteros. Determinar la mínima cantidad de elementos de s tales que al ser eliminados de la secuencia, el resto de los elementos forman una secuencia unimodal.

   `Tp2Ej1.in`: Contiene un conjunto de instancias a tratar, cada una almacenada en una línea diferente del archivo, donde aparece n seguido de $s_1, s_2, \ldots s_n$. El archivo termina con una línea donde n vale -1, la cual no debe interpretarse como una instancia de entrada.

   `Tp2Ej1.out`: Contiene una línea por cada instancia de entrada, donde aparece la cantidad pedida.

2. El gobierno está planeando la construcción de una nueva ciudad para mudar ahí la capital del estado. El proyecto está casi completo. Lo único que falta es decidir la forma de circulación dentro de la ciudad. La ciudad está formada por esquinas y calles. Cada calle conecta entre sí determinado par de esquinas. El gobierno quiere asignarle una dirección de circulación a cada calle de modo tal que los vehículos puedan ir de cualquier esquina a cualquier otra. Decidir si esto es posible.

   `Tp2Ej2.in`: Contiene un conjunto de instancias a tratar, cada una almacenada en varias líneas del archivo. Cada ciudad de n esquinas se representa con $n + 1$ líneas. Cada esquina se identifica por un número correlativo entre 1 y n. En la primera línea aparece n. En la i-ésima de las n líneas restantes aparece la cantidad de esquinas conectadas a la i-ésima esquina, seguida de la lista de esas esquinas. El archivo termina con una línea donde n vale -1, la cual no debe interpretarse como una instancia de entrada.

   `Tp2Ej2.out`: Contiene una línea por cada instancia de entrada, la cual dice "`fuertemente conexo`" si es posible realizar una asignación como la descripta, o dice "`no`" en caso contrario.

3. Bernardo Félix Souza finalmente fue a la cárcel. Su operación de lavado de dinero fue descubierta, sus grupos especiales de trabajo fueron desarticulados, y su pequeño islote en el Pacífico Sur fue expropiado y luego vendido en subasta pública. Bernardo es considerado un criminal muy peligroso, por lo que fue enviado a una prisión de máxima seguridad de la cual no es posible escapar por la fuerza. La prisión está formada por n habitaciones conectadas por m pasillos. Cada pasillo conecta exactamente dos habitaciones y puede ser transitado en ambas direcciones. En toda la prisión hay p puertas indestructibles, cada una de las cuales puede abrirse con una única llave. Tanto las puertas como las llaves están repartidas en las habitaciones, de tal forma que cada habitación puede tener una única puerta o almacenar una única llave, pero no ambas cosas. Si una habitación tiene puerta, la llave correspondiente es necesaria para entrar a la habitación, independientemente del pasillo que se use para llegar a la misma. Bernardo sobornó a un guardia para que le consiguiera el mapa de la prisión. En el mapa aparecen las habitaciones, los pasillos, y la ubicación de las puertas y de las llaves. Según el mapa, Bernardo se aloja en la habitación identificada con el número 1, mientras que desde la habitación número n es posible salir de la prisión. Decidir si Bernardo puede recorrer las habitaciones recolectando llaves y abriendo puertas, de manera tal de llegar a la habitación número n y así escapar.

   `Tp2Ej3.in`: Contiene un conjunto de instancias a tratar, cada una almacenada en varias líneas del archivo. Cada prisión de $n \geq 4$ habitaciones, m pasillos y p puertas se representa con $1 + p + m$ líneas. Cada habitación se identifica por un número correlativo entre 1 y n. En la primera línea aparecen n, p y m. Cada una de las p líneas siguientes describe una puerta y su correspondiente llave, indicando en qué habitación está la llave y en qué habitación está la puerta que esa llave abre. Asumir que las habitaciones 1 y n no tienen ni puerta ni llave. Cada una de las m líneas siguientes describe un pasillo, indicando las dos habitaciones que el pasillo conecta. Asumir que no hay más de un pasillo que conecte el mismo par de habitaciones. El archivo termina con una línea donde n, p y m valen -1, la cual no debe interpretarse como una instancia de entrada.

   `Tp2Ej3.out`: Contiene una línea por cada instancia de entrada, la cual dice "`libre`" si Bernardo puede escapar, o dice "`no`" en caso contrario.